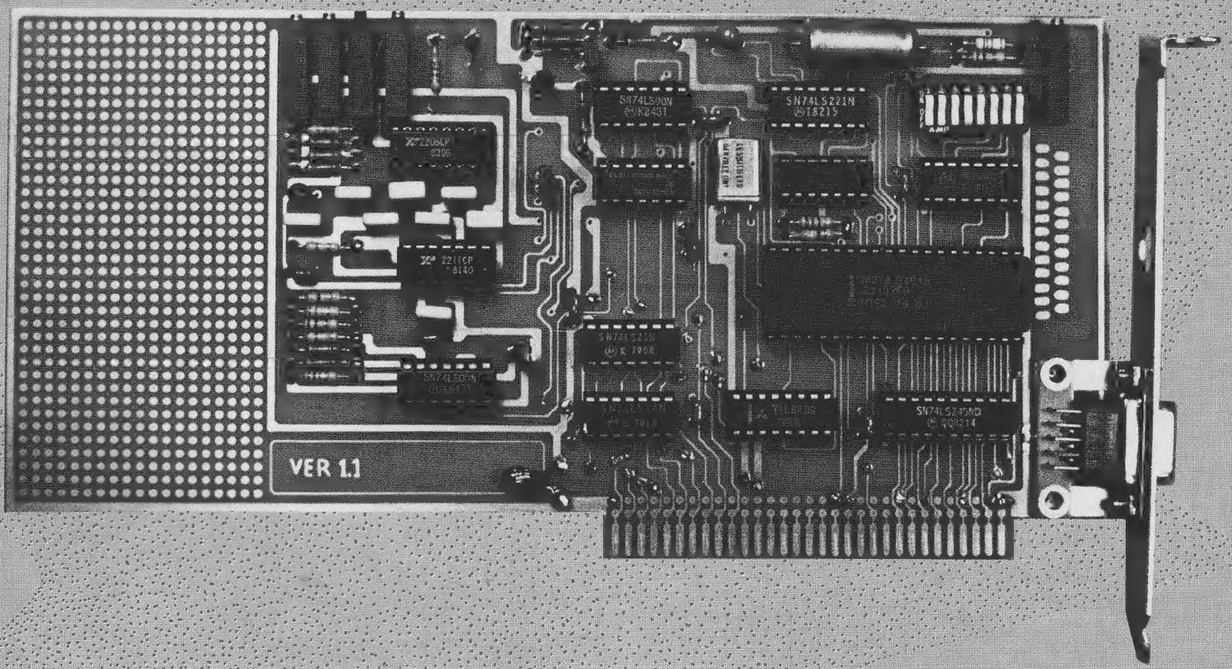
AUGUST 1986 / $2.50

PACKET RADIO

ham radio magazine

ham radio • August 1986

from Canada—



TNC for the IBM PC

hr

focus on communications technology

a packet radio TNC for the IBM PC

Single plug-in board extends usefulness of popular pc to digital communications

The advent of the IBM Personal Computer, with its open and well-documented architecture, has made it possible for hobbyists to extend the use of the microcomputer into many varied and challenging areas. This article describes a complete packet radio terminal node controller which can be implemented on a single circuit board that plugs into one of the slots in the PC. All that's required to get on the air is to connect up an Amateur Radio transceiver (**see fig. 1**).

the TNC: basic packet building block

Traditionally, packet radio has been achieved through the use of a device called a *terminal node controller*, or TNC. A TNC consists of a processor (such as the 8085 or Z80), a serial or parallel port to connect the TNC to a terminal or microcomputer running a terminal emulation program, and a synchronous port for the communications channel. A certain amount of RAM (random access memory) is available, and the necessary programming is provided on ROM (read only memory).

Because a TNC contains many components that are already present in our microcomputers, merging the function of the TNC into the microcomputer is an effective way to reduce the cost of a packet radio system. With suitable programming, the function of the TNC can be implemented in the microcomputer at a fraction of the cost of a separate TNC. There's a bonus, too, in that a higher level of function is available to the user because of the close coupling of the channel to the host processor.

In this implementation, an Intel 8273 programmable SDLC/HDLC protocol controller chip is interfaced to the bus in the IBM PC. This can be built on a prototype card or on a custom-printed circuit board. It's also possible to construct, on the same card, a simple but effective 1200-bps Bell 202-compatible half-duplex modem, a device in widespread use among packet radio enthusiasts. Software that runs the standard AX.25 packet radio protocol is available. The adapter is called the HAPN-1 adapter; HAPN stands for "Hamilton and Area Packet Network," the name of our packet radio club.

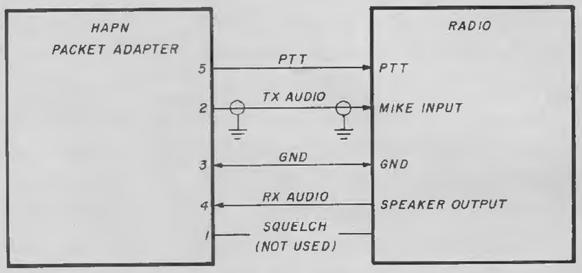


fig. 1. Simple radio hookup via microphone plug and speaker output. Note: Jumper J2 should be installed and J1 removed on adapter board.

hardware

The circuit shown in **fig. 2** uses an 8273 (U6) interfaced to the PC bus and a 1200-bps modem which can be connected to an Amateur Radio transceiver. The

By Jack Botner, VE3LNY, Ron Bradshaw, VE3IUV, Max Pizzolato, VE3DNM, and **John Vanden Berg, VE3DVV**, Hamilton and Area Packet Network, Box 4466, Station D, Hamilton, Ontario, Canada L8V4S7.

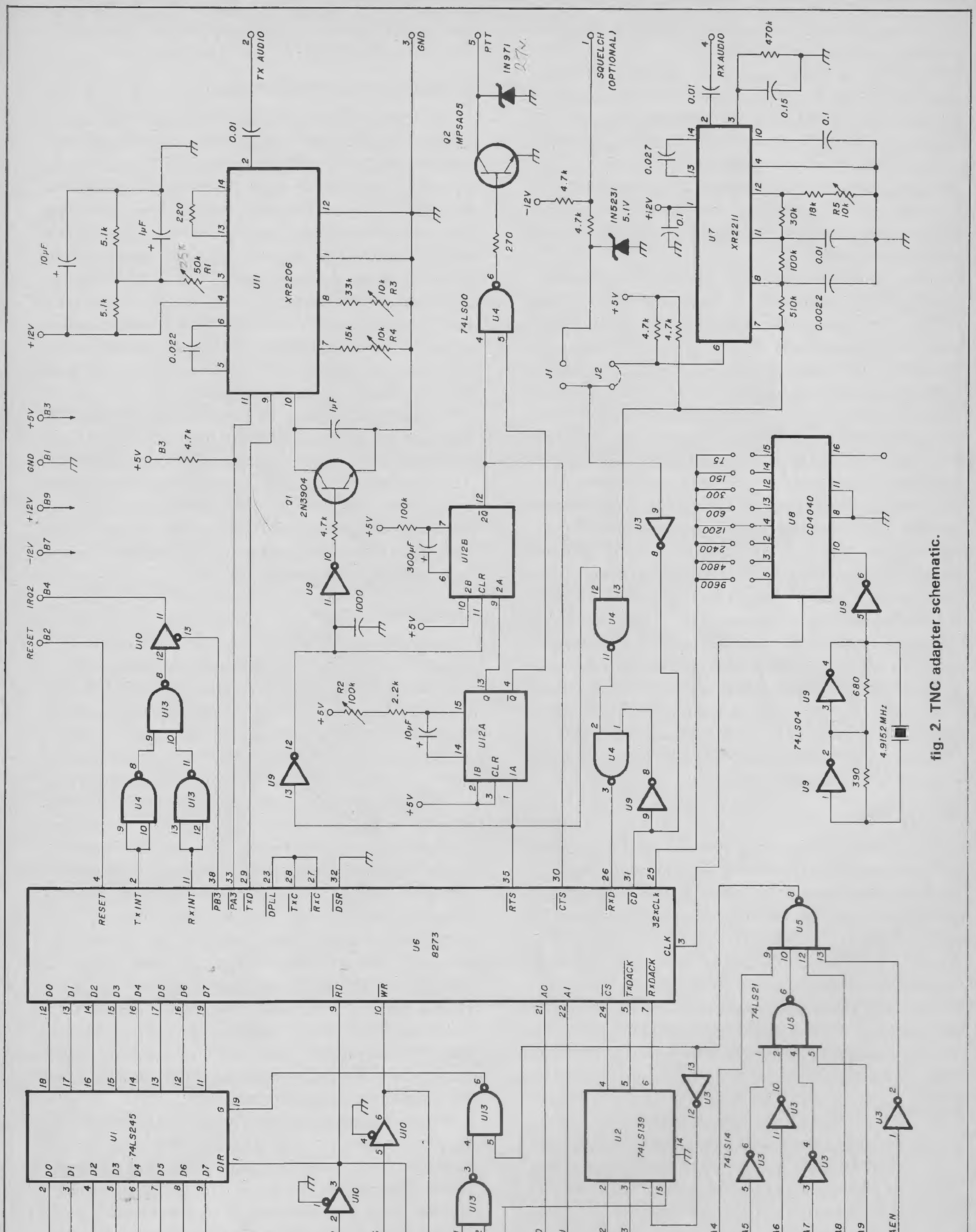


fig. 2. TNC adapter schematic.

8273 chip, called the *protocol controller*, is the most important part of the adapter.

The 8273 data lines are connected to the PC bus using a 74LS245 three-state buffer U1. The state and direction of the 74LS245 are controlled by the I/O Read and I/O Write lines and the I/O decode logic. The I/O decode circuit, made up of U2, part of U3, and U5, responds to addresses in the range of 310 through 31F (hexadecimal notation). The 8273's address requirements are met by a 74LS139 two- to four-line decoder (U2).

Data transfers to and from the 8273 may be done using polling, interrupts, or Direct Memory Access (DMA). This design uses interrupts, because they're easy to implement and permit a form of background operation that doesn't require running the packet application program constantly. The 8273 provides separate interrupt signals for transmit and receive (Tx Int and Rx Int, pins 2 and 11, respectively). The two interrupt lines are ORed together (part of U13) so that the adapter uses only one of the PC's hardware interrupt lines. The software can easily distinguish between transmit and receive interrupts by reading the 8273 status register. The combined interrupt line is buffered to the PC bus with a 74LS125 three-state bus buffer (part of U10) and is wired to the PC's IRQ2 interrupt line. IRQ2 was chosen because it is the least frequently used interrupt line in the PC. The circuit allows the interrupt line to float on the bus until the 8273 is initialized and the PB3 control port on the 8273 is activated. This means that the card can share IRQ2 with other hardware as long as the other device also floats its IRQ2 line when not in use and the two devices are not used at the same time.

clock signal

The 8273 features a digital phase-locked loop which makes it possible to derive the synchronous clock signal from the receiver data stream, a feature that greatly simplifies the design of the hardware. Pins 27 and 28 of the 8273 (Tx Clock and Rx Clock) are tied together to pin 23 (DPLL output). The 32X clock signal is provided by a 4040 binary counter (U8), giving a selection of data rates from 75 to 9600 bps by jumper or dip switch. A 4.9152-MHz crystal oscillator (U9) provides the clock signal for the 8273 and U8, the baud rate divider.

modem

The modem uses a pair of Exar chips, the 2206 function generator (U11) and the 2211 FSK demodulator (U7). The 2206 generates the two tones, 1200 and 2200 Hz, used for transmit; the 2211 decodes the two tones from the receiver audio. Two control signals are required by the 8273: Carrier Detect (CD, also known as DCD or RLSD in various contexts), indicating that the radio channel is busy, and Request to Send (RTS), indicating that the 8273 wants to transmit.

channel busy indication

The CD signal (channel busy indication) can be derived two ways, either internally by using the data carrier detect (DCD, pin 5) from the 2211 or externally by using the squelch from the receiver. Using the squelch to indicate the radio channel busy condition is superior but usually requires modifications to the radio. It guarantees, however, that the system will not transmit when another signal — possibly voice or some other non-data signal — is present on the channel. The internal DCD from the 2211 signals only when actual data is being received. This is the recommended way when no squelch line is available or the QRM level is high, such as 20-meter operation.

To use the internal DCD, install jumper J2. The 2211 checks the incoming signals for data and signals DCD to the 8273. To use the squelch CD, install jumper J1. The two resistors and zener diode convert the squelch signal from the radio to a TTL level. The signal is debounced by using a 74LS14 Schmidt trigger inverter gate (part of U3). An example of the squelch pickup from the radio is given in **fig. 3**.

transmit mode

To transmit, the 8273 brings up RTS, which triggers U12a (CTS-delay single shot) and brings up PTT via U4 pin 5 (negative or gate) and the 2MPS-A05 switching transistor (Q2). The U12a single shot allows the radio to stabilize when going into transmit before sending data; this function is called the *clear-to-send delay*. When U12a times out, the watchdog timer U12b starts, bringing up the CTS line for the 8273. This signals the 8273 to send the data. At the same time, it continues to activate the PTT line via U4 pin 4. When the 8273 is finished transmitting it drops RTS, which applies a reset to the watchdog single shot, causing it to drop the PTT line. If for any reason the RTS line stays up too long (as in a crash of the system, for example) the watchdog times out after about 22 seconds and shuts down the transmitter, releasing the radio channel to other users. Transistor Q1 enables the tones on the 2206 only when RTS is up (i.e., when 8273 is transmitting), so that there's no interference audio from the modem when the microphone is used for normal voice conversation.

Figure 1 shows how the adapter can be hooked up to a handheld radio. The TX-audio line is connected to the microphone audio input by means of a shielded cable. The receiver audio and PTT are taken from the speaker plug. Remove jumper J1 and install it at J2 to use the DCD line from the demodulator. For best performance, adjust R1 (a 50 k trimpot) for about 3-kHz deviation.

Integrated circuit voltage and ground pinouts.

IC	Type	Gnd	+5	+12
U1	74LS245	10	20	—
U2	74LS139	8	16	—
U3	74LS14	7	14	—
U4	74LS00	7	14	—
U5	74LS21	7	14	—
U6	8273	20	40	—
U7	XR2211	4	—	1
U8	CD4040	8	16	—
U9	74LS04	7	14	—
U10	74LS125	7	14	—
U11	XR2206	12	—	4
U12	74LS221	8	16	—
U13	74LS00	7	14	—

TNC to 2-meter radio interconnections

Figure 3 shows a hookup to an IC22s 2-meter radio. Here the TX-audio is inserted after the audio pre-amp, Q29, in the IC22s. If a DTMF (touchtone) input is available, it could be used instead. A 390-ohm resistor takes off the RX-audio before the volume control. The PTT goes to the front panel microphone connector. Transistor Q100 inverts and buffers the radio squelch. All the connections are brought out via the utility connector on the back of the radio. An audio output of about 1000 mV will give you about 3-kHz deviation on transmit. Remove jumper J2 install it at J1 on the packet adapter board. Set the CTS delay (R2) for about 300 msec; if it's set too short, the beginning of the transmitted frame will be missing and therefore impossible to copy. If it's too long, time will be wasted transmitting idle flag characters.

combining TNC with the host computer simplifies system design

In the traditional TNC, the programming is supplied on a ROM IC chip, and is relatively invisible to most users. In the HAPN implementation, the software is supplied as programs on a diskette, which are run like any other program on the PC.

One of the limitations of the traditional TNC is the loose coupling, or distance, between the TNC and the microcomputer acting as the terminal. By necessity the TNC strips the packet data down to the most elemental level, that of a serial stream of characters. This makes it difficult for the host computer to learn anything about what's going on in the TNC at the packet level. On top of this is placed a layer of commands, escape characters, and flow control, making even the best of packet host programs appear clumsy and unfriendly.

But including the packet hardware in the PC itself overcomes all of these problems. The programmer is free to design the software — particularly the end-user interface — in any way desired, without the need to compromise for the TNC interface. The result is a high-function packet system with a friendly user interface.

HAPN packet software implementation

In order to make effective use of the 8273, the following must be included:

- a BIOS (Basic Input/Output System)-like hardware driver that can be loaded as an extension to the system to drive the 8273;
- a protocol manager with an application program interface; and
- an application program that is run at the discretion of the user.

In addition, the 8273 must be made to operate in the background so that the node represented by the 8273 can be active all the time.

These requirements can be met by developing the software in at least two programs. The first program, M25, contains the hardware driver, protocol manager, and application program interface. The second program, C25, makes up an end-user application program. Since the 8273 operates on interrupts, no application program assist is necessary to receive or transmit data. Total independence from the application program has been achieved by using the PC's timer tick hardware interrupt exit as the protocol manager's dispatcher for scheduling events.

M25, the hardware driver module, contains a number of distinct functions and is made up of several routines. These routines have been link-edited to make up one program, which is run once after the system has been booted to become a resident operating system extension.

The application program, C25, contains a set of end-user functions primarily concerned with the sending and receiving of data. It can be used to activate the hardware functions in M25 via the application program interface. C25 would be run any time the user wishes to access the packet node; otherwise the PC can be used to run other application programs.

The main function of M25 is to service interrupts from the 8273, so that data can be transferred into and out of the channel. The interrupt handling routine is divided into two parts, one that handles transmit and one that handles receive interrupts. During its operation the 8273 generates interrupts for each byte transmitted or received, as well as for various transmit and receive frame complete events. Also a variety of error conditions may be reported to the processor — for example, when a frame has been received with an invalid CRC (cyclic redundancy check) field, generally caused by noise.[1]

The receive interrupt handler stores each byte of a frame in a buffer. When the frame is complete the 8273 generates another interrupt. At that point the frame may be inspected to make sure it's free of errors and is valid by the rules of protocol in use. Frames that

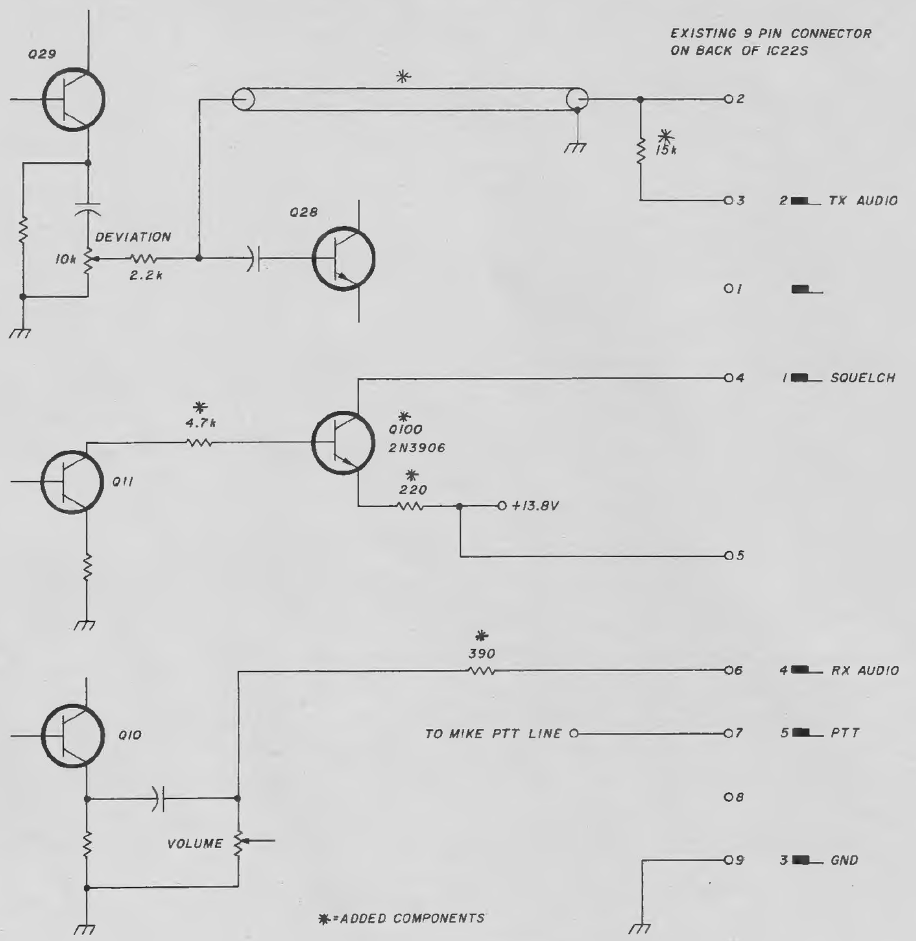


fig. 3. Example of squelch pickup.

contain errors are simply discarded. If a valid frame is received, some link-level protocol management is performed on the frame before exiting the interrupt routine. An example of this processing would be to look for connect requests and queue a response when one is received. It's also possible to delete request frames from the transmit queue after a valid response has been received.

The transmit interrupt routine passes each byte of the frame from a buffer to the 8273 as required. When the frame has been completely transmitted, the 8273 generates an early frame complete interrupt. At this time the software can decide if another frame should be transmitted in the same packet. In the AX.25 protocol, up to seven frames may be transmitted in one packet. If another frame is available, a command is issued to the 8273 to transmit another frame during the current transmission. The 8273 automatically inserts a byte of flags between frames sent in this way.

When the packet is complete the 8273 generates a transmit complete interrupt. The software uses this interrupt to do some protocol management on the transmit queue. For example, because request frames may have to be retransmitted at a later time, they're left on the transmit queue until they're acknowledged; response frames, on the other hand, are transmitted only once and may be deleted from the transmit queue at this time.

In this implementation an OR gate is used to drive both transmit and receive interrupts on the same IRQ line to the PC's 8259 interrupt controller. M25 distinguishes between them by reading the 8273's status register, which contains two bits that specify which type of interrupt was generated. It's interesting to note that because of noise on the channel, the 8273 occasionally becomes confused and generates interrupts that cannot be identified by the status register. When this kind of error occurs, M25 issues a software reset command and then re-initializes the 8273 to ensure correct operation. Statistics are kept on the various events that occur in the packet system, and may be displayed by the user.

TNC adapter parts list (fig. 2).

R1	50k trimpot
R2	100k trimpot
R3	10k trimpot
R4	10k trimpot
R5	10k trimpot
R6	5.1k
R7	5.1k
R8	220
R9	15k
R10	33k
R12	2.2k
R13	100k
R14	1k
R11,15,16,18,19	4.7k
R17	470k
R20	510k
R21	100k
R22	30k
R23	18k
R24	390 ohms
R25	680 ohms

All resistors 5 percent 1/4 watt
10-turn trimpots recommended for R3, R4, and R5

CR1	1N5231 (5.1 volts 5 percent zener)
Xtal	4.9152 MHz
Q1	2N3904
Q2	MP5-A05
J1/J2	2 pin jumper connector
J3	8-position DIP switch
DB-9	(male) printed circuit board connector
U1	74LS245
U2	74LS139
U3	74LS14
U4	74LS00
U5	74LS21
U6	8273
U7	XR 2211
U8	CD 4040
U9	74LS04
U10	74LS125
U11	XR 2206
U12	74LS221
U13	74LS00

I.C. Sockets:	Seven 14-pin
	Four 16-pin
	One 20-pin
	One 40-pin

C1	10 μF tantalum
C2	330 μF tantalum
C3	.001 μF
C4	10 μF tantalum
C5	.022 μF
C6	1 μF tantalum
C7	.01 μF
C8	1 μF tantalum
C9	.1 μF
C10	.15 μF
C11	.027 μF
C12	.0022 μF
C13	.01 μF
C14	.1 μF
C15	.01 μF

M25 contains a dispatcher routine that runs periodically. Its purpose is to manage timeouts and initiate events as they're required by the protocol. The timer tick interrupt, which is issued every 55 milliseconds on the PC, is a convenient way to give control to the dispatcher for this purpose. The dispatcher monitors the channel for activity and initiates frame retransmissions when timeouts occur. A random number generator produces the timeout value so that repeated collisions with another station can be avoided. The dispatcher also keeps track of the number of retransmissions and produces a loss of contact condition when a preset limit is reached. Because it's driven from the timer hardware interrupts, the dispatcher runs independently of any programs on the PC.

M25 also contains the application program interface. This interface allows a program running on the PC to access the 8273 channel. Interface functions include open, read, write, inquire, modify, and close. Note that the application program issuing the open does not have to close before terminating; this means that the packet node will continue to operate regardless of what program is run next on the PC. An alarm feature is also included so that the user is alerted if a connect request is received. The application program interface uses a software interrupt, level 84 (hex), for access by programs.

M25 includes an installation routine for loading itself into memory and becoming resident. Storage is reserved for transmit and receive buffer pools and a trace table. The application program interface vector is loaded so that the 8273 driver interface can be accessed. However, packet operation doesn't begin until it's initiated by an open request from an application program such as C25.

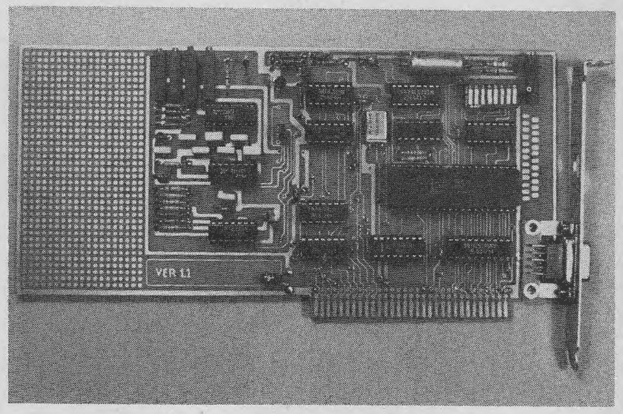


fig. 4. HAPN adapter board plugs into IBM PC slot.

C25, the end-user application program, is an ordinary DOS program designed as an interface between the user (keyboard, display) and the packet channel. It tests to see if the 8273 driver is installed and issues various requests to the 8273 driver application program interface.

data entry

The user of a packet node is mainly interested in sending and receiving data. This data may be entered through the keyboard or read from a disk file. Received data is displayed on the screen and may be saved to a disk file if desired. Other control functions include initiation of a connect request and specification of repeater linking information.

viewing the screen

The screen is divided into three areas: one for displaying received data, another for displaying typed input, and a third for indicating status. The receive data display area, the largest of the three, scrolls when it fills up. The typed-input area is two lines (160 characters) long. No data is transmitted until the return key is pressed, and a comprehensive set of editing functions is provided so that the user gets an opportunity to change the typed data before it's transmitted. The status area shows information such as lock key status, active function keys, and node callsign and linking repeater information.

other user-friendly features

C25 includes several convenient features. One, the scroll lock key, temporarily stops the screen from scrolling. Another automatically generates, upon request, a test message containing your callsign, the date, and time.

C25 also contains a self-customizing routine so that the program options can be set by each user. Program options are stored inside the program module rather than in a separate file, making C25 more convenient to run. A pop-down menu is provided for entering and displaying repeater information.

There are two exit options from C25: one issues a close to terminate packet operation and one does not. In the latter case the packet node is left operational while other programs are run on the PC. When C25 is run again later it will pick up the status of the packet node using an inquiry call to the application program interface.

Two utility programs are provided in addition to the basic functions described above. S25 formats either the packet trace table or the packet statistics counters. The trace table is useful for analyzing events such as protocol violations, in which two nodes appear to misbehave for no apparent reason. T25 is a test program with a number of functions for testing and setting up the 8273 adapter. It can be used to test the 8273 and bus interface hardware and to perform adjustments on the modem.

building the adapter

The adapter shown in **fig. 2** can be constructed on a readily available prototype board for the IBM PC. Consult the technical reference for details of the board's edge connector.[4] Although any suitable connector may be used, a DB-9 male connector was used for the transceiver interconnection. The only critical components are the frequency control trimpots in the modem, R3, R4, and R5. It's a good idea to use multiturn trimpots for these components. You should also use stable mylar or polyester capacitors in the circuits around the 2206 and 2211.

When connecting the adapter to a transceiver, be careful to use good quality shielded wire for the transmit audio lead. It's also a good idea to bypass each end of the cable with a 0.001 μF capacitor and place a ferrite bead over the center conductor at each end. This will prevent RF pickup, which can cause erratic operation of the system and transmission of garbled signals, into the transmit audio circuit of the transceiver.

It's good practice to bypass the +5-volt supply lead to each IC with a 0.1 μF tantalum capacitor. Otherwise, layout and wiring are not critical, if good construction practice is used. For those who prefer not to wire their own boards, both bare boards and assembled and tested boards are available from HAPN.

adjusting the adapter

After the board has been assembled and checked for shorts, it's ready for testing. The first test is an internal loopback test of the 8273 using T25.EXE, the test program supplied with the software. This tests the data buffers (U1), the 8273, address decoding, and the clock circuit.

Now adjust the modem. Start by setting up the two transmit frequencies, F1 and F2. Float U11 pin 10 to enable the tones. Then float U11 pin 9 and adjust R3 for 1200 Hz (use a frequency counter or an accurate scope). Next ground U11 pin 9 and adjust R4 for 2200 Hz. If the output is too low for the frequency counter, adjust the level control, (R^1).

Next adjust the CTS delay by putting a scope on U12 pin 13 and key up the transmitter (using C25.EXE). Adjust R1 for 100-500 milliseconds, depending on how fast your rig switches from receive to transmit; a value of 300-400 milliseconds works well for most rigs.

Demodulator adjustment is a bit more tricky. You have to jumper the connecter pin 2 (Tx audio) to pin 4 (Rx audio) and float U11 pin 10 to enable the tones. When C25 has initialized the 8273 it will transmit flags as 1's and 0's. Adjust R5 so the waveform at U7 pin 7 looks the same as on U11 pin 9.

The remaining adjustment is of the audio level (R1) going to the radio. The level varies widely with the type of hookup into the radio. Adjust this level so the deviation is approximately 3 kHz on the transmitter. This completes the adjustments.

a note on HAPN

The following items may be ordered from HAPN. All prices are in United States dollars and include postage and handling.

- Software only (four programs and documentation), $40.
- Software and bare board (includes set-up program and assembly instructions), $75.
- Software and assembled and tested board (see **fig. 4**), $199.

Please add $3 for overseas orders.

The HAPN adapter will run on any IBM PC, XT, AT, or compatible computer with at least 128K memory and a display capable of 80-column text. The card is 8.5 inches (21.6 cm) long.

Orders should be addressed to HAPN, Box 4466, Station D, Hamilton, Ontario, Canada, L8V4S7.

Dedicated to furthering the state of packet radio, HAPN is currently involved in the design of a station node controller. The station node controller is capable of unifying a packet radio network by providing message routing, gateways, and high-speed links to other station nodes.

HAPN is also involved in experimentation with 4800-baud modems and with V3, the newly developed VADCG protocol. HAPN has experimental V1 and V2 software that runs on the adapter described in this article; for more information, contact HAPN at the above address.

references

1. Intel "8273, 8273-4, 8273-8 Programmable HDLC/SDLC Protocol Contoller," *Peripheral Design Handbook*.
2. J. Botner, "A Packet Radio Adapter for the IBM PC." *QEX: The ARRL Experimenter's Interchange*, January, 1985.
3. T. Fox, "AX.25 Amateur Packet Radio Link Layer Protocol," Version 2.0, October 1984. American Radio Relay League.
4. IBM *Technical Reference*, Personal Computer Hardware Reference Library.

ham radio